CS 6200 Information Retrieval

Seattle/Fall 2019

Assignment 1 Web Crawling:
The Karen Sparck-Jones edition

Announced Sep 16, 2019

Due: Sep 30, 9am

Your task in this assignment is to implement a simple web crawler and apply it to crawl a portion of Wikipedia. The goal is to get you better aware of (some of) the work involved in crawling the web. In addition, once you have a crawler working, there's a lot of interesting projects you can do on the web.

Inputs and Outputs

1. The input parameters to the crawler will be
   a. a single seed URL **SeedUrl**,
   b. number of pages to be crawled **Numpages.**

2. The outputs to be submitted will be as follows:
   a. One or more **Python program files** which implement this assignment.
   b. A small (shell) script named **RunCrawler** (with an appropriate file extension like .sh , .py, etc.) which takes in the parameters SeedUrl and Numpages as inputs to create the desired outputs.
   c. A text file named **URLsCrawled.txt** which lists the URLs of the (up to) Numpages URLs that were crawled. This file should contain (exactly) one URL per line.
   d. A file named **stats.txt** with the maximum, minimum and average size of the files crawled, and the maximum depth (see below for definition of 'depth') you reached when you stopped. To keep it simple, use the following format:
      Maximum size: nnnnnnn bytes
      Minimum size: mmmm bytes
      Average size: aaaaaa bytes
      Maximum depth reach: d
   e. A file named **README.txt** which contains:
      i. A one or two sentence answer to the question: What was the most difficult part of this assignment?
      ii. Optional: Any additional information you may wish to provide about running the script **RunCrawler**.

What's expected to be done

3. Starting with the seed URL as depth 1, crawl each URL to a maximum depth of 5. That is, consider that the pages linked from the seed page are depth 2, the pages they linked to are depth 3 etc. Do not go beyond depth 5.
4. The crawler should stop when you have crawled **Numpages** unique pages (URLs) or if you are going beyond the maximum depth of 5.
5. **You may use standard libraries to get the contents of a document via HTTP, and follow HTTP redirects. (**So, it's OK to use urllib and re, but not BeautifulSoup or requests.)
6. However, *you must write your own code* to extract the links from each page, and to keep track of the pages you have crawled.
7. **You must also use politeness rules and wait at least one second between requests to the web server to get new pages.**
8. At all times, you must have a list (frontier) of URLs to be crawled next.
9. Because you know you will be crawling Wikipedia, you can optimize the crawling. Use the following rules to include or ignore links:
   a. Remember that web pages or web sites often use a base URLs, and further URLs within the web page/web site may start as relative URLs, say with "/wiki/…."
   b. *Include only links that start with http://en.wikipedia.org/wiki/* This way you will only be considering Wikipedia articles in the English language, and will ignore other pages (which may or may not be HTML pages). **Note that the protocol may be http or https.**
   c. Ignore links to http://en.wikipedia.org/wiki/Main_Page . Most wiki pages have a reference to the main page, which we can ignore.
   d. Ignore links with a colon (':') after the host name. That is, ignore links like https://en.wikipedia.org/wiki/Help:Authority_control which are specific to Wikipedia.
10. To keep things simple, write code to retrieve pages one by one (in a single-threaded fashion). When you have retrieved a page, get its size (from the HTTP response, or by other means) and use it to calculate size statistics. Keep updating the file URLsCrawled.txt as you crawl each URL. At the end of crawling, compute and write out the stats.txt file.
11. Make sure the code is adequately commented.
12. When the coding is completed, run the program with the following values of parameters:

    SeedUrl = https://en.wikipedia.org/wiki/Karen_Sp%C3%A4rck_Jones

    and Numpages = 1000
13. One way to do this: as you crawl the URLs, save all the files crawled in a folder, with each file having the filename nnn.txt where nnn is a number from 1 to Numpages.
14. Be sure to follow the instructions above. You may lose points if you miss any of these instructions. Also, keep in mind the academic integrity rules we discussed.

Important note: We will be building upon this code and using the output of this code in the next assignment(s). So, retain the code, and save the crawled documents as well.